UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
09/604,987	06/28/2000	Srivatsan Parthasarathy	MS146910.1	6447

27195   7590   07/18/2003

AMIN & TUROCY, LLP
24TH FLOOR, NATIONAL CITY CENTER
1900 EAST NINTH STREET
CLEVELAND, OH 44114

EXAMINER
VU, TUAN A

ART UNIT	PAPER NUMBER
2124	5

DATE MAILED: 07/18/2003

Please find below and/or attached an Office communication concerning this application or proceeding.

PTO-90C (Rev. 07-01)

	Application No.	Applicant(s)
Office Action Summary	09/604,987	PARTHASARATHY ET AL.
	Examiner	Art Unit
	Tuan A Vu	2124

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE _3_ MONTH(S) FROM THE MAILING DATE OF THIS COMMUNICATION.
- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If the period for reply specified above is less than thirty (30) days, a reply within the statutory minimum of thirty (30) days will be considered timely.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133).
- Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on _09 May 2003_.

2a)☐ This action is **FINAL**.   2b)☒ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) _1-35_ is/are pending in the application.

     4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☒ Claim(s) _10-17 and 22_ is/are allowed.

6)☒ Claim(s) _1-8,18 and 23-35_ is/are rejected.

7)☒ Claim(s) _9, 19-21_ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☒ The drawing(s) filed on _28 June 2000_ is/are: a)☒ accepted or b)☐ objected to by the Examiner.

     Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

11)☐ The proposed drawing correction filed on _____ is: a)☐ approved b)☐ disapproved by the Examiner.

     If approved, corrected drawings are required in reply to this Office action.

12)☐ The oath or declaration is objected to by the Examiner.

Priority under 35 U.S.C. §§ 119 and 120

13)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

     a)☐ All b)☐ Some * c)☐ None of:

         1.☐ Certified copies of the priority documents have been received.

         2.☐ Certified copies of the priority documents have been received in Application No. _____ .

         3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

     * See the attached detailed Office action for a list of the certified copies not received.

14)☐ Acknowledgment is made of a claim for domestic priority under 35 U.S.C. § 119(e) (to a provisional application).

     a) ☐ The translation of the foreign language provisional application has been received.

15)☐ Acknowledgment is made of a claim for domestic priority under 35 U.S.C. §§ 120 and/or 121.

Attachment(s)

1) ☒ Notice of References Cited (PTO-892)

2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)

3) ☐ Information Disclosure Statement(s) (PTO-1449) Paper No(s) _____ .

4) ☐ Interview Summary (PTO-413) Paper No(s). _____ .

5) ☐ Notice of Informal Patent Application (PTO-152)

6) ☐ Other: _____

DETAILED ACTION

1.   This action is responsive to the Applicant's response filed May 9, 2003.

Claims 1-35 have been submitted for reconsideration and are pending in the office action.

Claim Rejections - 35 USC § 103

2.   The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all

obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in
> section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are
> such that the subject matter as a whole would have been obvious at the time the invention was made to a person
> having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the
> manner in which the invention was made.

3.   Claims 1-8, 18, and 23-35 are rejected under 35 U.S.C. 103(a) as being unpatentable over

Evans et al., USPN: 5,805,899 (hereinafter Evans), in view of Buxton, USPN: 6,182,279

(hereinafter Buxton).

    As per claim 1, Evans discloses a method for integrity checking employable by

application programs at runtime (e.g. Fig. 2d), such method comprising:

    providing an assembly with manifest (col. 4, line 60 to col. 5, line 2; *Executable Linking*

Format (ELF) Object file, Fig. 5, 10) that contains a list of versioned objects (e.g. *Relocatable*

object 118, Shared object 114, Fig. 2c-d; *versioned shared objects*, col. 9, lines 20-22) that make

up the assembly (e.g. *ELF header, Section 1 ... Section N* - Fig. 10 – Note: listed sections

representing each object is equivalent to list of objects contained in the manifest of assembly);

    providing a manifest with a hash value of at least one objects of the list of sections (e.g.

Section 506, Fig. 5 and *Hash 614*, Fig. 6; *Section Header, Hash* - Fig. 11 – Note: the header of

each sections making up the ELF file with a hash of the version name is equivalent to manifest

having hash of objects of the section list).

But Evans does not specify that the list of objects making up the assembly via the manifest is a list of modules nor does Evans specify that the hash of the object versions of the list of objects/sections is a hash of the contents of at least one modules of such list. Evans, however, teaches linking of relocatable objects or compiled files (Fig. 1; col. 6, lines 30-38), hence suggests a dynamic linking of compiled modular components or source files converted into objects. Evans further shows a dependency linking means to interrelate object versions making up the manifest of the assembly (*version dependency* – Fig. 10; *next version dependency section* – Fig. 11; Fig 16), i.e. the contents of such list as well as their hashed value representation are expressed or pointed to via such ELF file or assembly manifest during the objects or modules runtime loading and linking. The hash representation of interlinked sections being equivalent to contents of section/object listed in the manifest has been thereby suggested. Buxton in a system to load objects in an object-oriented programming environment using *templates* (e.g. *DLL storage 205, OLE Libraries, Template 208* - Fig. 2 – Note: Loading dynamic libraries and linking them to applications is equivalent to runtime linking) or manifest analogous to the ELF file by Evans, with tracking of object identification (e.g. col. 12, lines 1-15; cols. 19-20) and integrity checking (e.g. Fig. 7-8B), discloses the hash of the contents of components making up *container* to be loaded via a template execution (e.g. Fig. 2; Fig. 3A; col. 9, lines 8-25) with each component being a self-contained module (e.g. col. 8, lines 55-63). It would have been obvious for one of ordinary skill in the art at the time the invention was made to implement the linking of inter-related object versions representing objects listed in the manifest during the integrity checking and dynamic linking by Evans such that it forms all related objects via the files linking process to be modules listed in such manifest and also includes a hash of the

contents of such modules just as taught by Buxton. The motivation is that it would make it easier to follow the interrelation of linked objects or files as taught by Evans by means of a structure that encompasses as much information as possible for change tracking and integrity checking of such linked files or objects converted into self-contained modules as suggested via the use of templates or manifest of assembly for dynamic loading by Buxton.

As per claim 2, in view of the combined teachings of Buxton and Evans from claim 1, the limitation of providing a hash of each module that constitutes the assembly would have been obvious because instead of just providing a hash of the module contents, matching a hash of each module for the integrity checking would fulfill the same purpose as matching a hash of a contents of such module, both methods equally serving identifying the module version or verifying its integrity at runtime.

As per claim 3, Evans further discloses providing identity information in the ELF file/assembly (Section *506*: fields *620*, Fig. 6; Section *510*: fields *1108, 1120, 1122*, Fig. 11).

As per claim 4, Evans discloses identity information including version information (e.g. Fig. 5, 6) in the manifest for the assembly of shared versioned objects, but fails to disclose the providing of the publisher information. Buxton, in a program modules runtime loading process analogous to that of Evans, discloses persistent storage representation of modules (re claim 1) in association with registration information indicating the provider or publisher (e.g. *vendor* – col. 10, lines 1-40; key *450C* - Fig. 3B). It would have been obvious for one of ordinary skill in the art at the time the invention was made to include a publisher or originator of the program components to assemble as suggested by Buxton to the assembling of versioned objects as disclosed by Evans because this would improve further the version-controlled linking process as

taught by Evans as more protection/security checking is applied to the sources

providing/publishing originating versioned objects or modified versions of objects.

As per claim 5, in reference to claim 1, Evans discloses hash of the section contents of

ELF file/assembly of versioned objects, but **fails to disclose** providing a hash of the contents of

such assembly. Buxton, in a method to provide dynamic application modules linking as

disclosed in claim 1, discloses providing a hash of the contents of a component at the end of the

components storage (e.g. Fig. 3A; col. 9, lines 8-25) used to form an assembly at runtime

loading. It would have been obvious for one of ordinary skill in the art at the time the invention

was made to propagate the modular contents hashing implementation to the level of the template

or assembly including all the modules to be loaded for execution and hashing of such assembly

contents at the end of the assembly as taught by Buxton so to implement such hashing to the

process of assembling versioned objects as disclosed by Evans. The motivation is that this

would add a higher level of integrity checking applied to the global representation of the

composing parts to the checking process as taught by Evans as more security is applied to the

each instance of dynamically assembling those modular objects as suggested by Buxton.

As per claim 6, Evans discloses weak/strong versions of objects to be linked by using the

analysis of global variables and pointing to another dependent versions (col. 7, lines 29-56; Fig.

8,15; col. 14, lines 32-41; Fig. 18 – Note: this is implicitly disclosing a hash matching process

equivalent to determining that one hash version, *weak version,* of objects to assemble is not the

latest or has been supplanted by another hash version, *non-weak version*, i.e. comparing hashes

values of the objects contents).

As per claims 7 and 8, in reference to claim 6, Evans discloses the checking of dependency of versioned objects to link (e.g. checking version information in the manifest – instant claim 8) and thereby determining which version is the latest update (Fig. 8, 15) but fails to disclose if the publisher of the assembly is (re claim 7) trustworthy via checking version information and publisher name information (re claim 8). In view of Buxton's teachings about registration information on the provider (publisher) of modules applicable to the linking process (see claim 4 above), it would have been obvious for one of ordinary skill in the art at the time the invention was made to include the trustworthiness checking of the publisher by Buxton as shown in claim 4 above to Evans's method of linking the most updated versions of objects for the same reasons as mentioned in claim 4.

As per claim 18, Evans discloses a computer readable medium (col. 2, line 66 to col. 3, line16) with an executable for a runtime application program, such medium comprising an assembly (e.g. col. 4, line 60 to col. 5, line 2; *Executable Linking Format* (ELF) Object file, Fig. 5, 10) including manifest containing a list of objects making up the assembly (e.g. *Relocatable object 118, Shared object 114*, Fig. 2c-d; *versioned shared objects*, col. 9, lines 20-22; *ELF header, Section 1 ... Section N* - Fig. 10); and a hash of at least one listed of objects in the assembly (e.g. *Section 506*, Fig. 5 and *Hash 614*, Fig. 6; *Section Header, Hash* - Fig. 11).

But Evans does not specify that such list of object is a list of modules, nor does Evans disclose the hash of objects is hash of contents one assembly of list of modules. But the list of modules limitation has been addressed in claim 1 using the combined teachings of Evans and Buxton and the limitation as to the hash of contents of assembly has been addressed in claim 5 above and is rejected herein using the same rationale therein.

As per claim 23, this is a system claim version of claim 1 above; hence is rejected herein using the corresponding rejection set forth therein.

As per claim 24, Evans discloses components to check versions dependency (e.g. components 126, 122, Fig. 2c-d) accomplishing version checking analogous to that addressed earlier in claim 6 above as far as being compatible with comparing of hash value of modules/versioned objects is concerned. Hence, the rejection in claim 6 herein applies.

As per claim 25, Evans further discloses identity and version information (Section *510*: fields *1104, 1108, 1112, 1120, 1122*, Fig. 11); a version dependency checker based on version information (col. 7, lines 29-56; Fig. 8,15; col. 14, lines 32-41; Fig. 18 – see claim 6), i.e. the matching of hash in the manifest against actual hash; but does not specify adopting an assembly based on the originator. But the limitation of the originator has been addressed in claim 4 using Buxton's registration of publisher information for checking integrity of loaded components at runtime, hence would have render the adopting of an assembly based on originator information obvious using the rejection in claim 4 and claim 6 combined.

As per claim 26, Evans further discloses recording and establishing of version dependency and inheritance (col. 2, lines 17-46; Fig. 2a-b; Fig. 8) and to check runtime version binding (Fig. 2c-d; Fig. 15) based on the version dependency records; and this is equivalent to disclosing binding policy.

As per claim 27, Evans discloses a method for facilitating integrity of assemblies employable by application program at runtime, such method comprising components for:

providing an assembly with manifest of assembly (col. 4, line 60 to col. 5, line 2; *Executable Linking Format* (ELF) Object file, Fig. 5, 10) that contains at least one referenced

sections (e.g. *section 1...n, dependency section 510* , Fig. 5) that make up the assembly (e.g.

ELF object file; Figs. 5, 10).

But Evans does not specify that the list of referenced sections above are referenced

assemblies; nor does Evans disclose providing the manifest with a hash of at least one referenced

assembly. .   In view of Buxton's teaching about generating a signature (hashing) of the modules

contents applicable to the loading/linking process (see claim 5 above) and Buxton's incremental

gathering of related containers (i.e. assemblies represented by templates) of classes during the

template execution (Fig. 6-7), it would have been obvious for one of ordinary skill in the art at

the time the invention was made to include hash the whole assembly of objects or representation

of such assembly (ELF file contents as taught by Evans or template by Buxton) and make it part

of the assembly hash dependency linking scheme as disclosed by Evans because this would

extend the security/version checking adopted by Evans on objects to the very container of such

versioned objects upon which version dependency applies, a form of trustworthiness checking as

has been shown in Buxton's loading process via registration and matching of incrementally

loaded classes (e.g. Fig. 6-7).

As per claim 28, see similar rejection in claim 24.

As per claim 29, this is analogous version of claim 26 in the context that the binding

policy herein applies to referenced objects; hence incorporates the same rejections of claim 26.

As per claim 30, Evans discloses a system for facilitating integrity of an assembly

employable by application program at runtime (Fig. 2d), such system comprising means for

relating a manifest (*Executable Linking Format* (ELF) Object file, Fig. 5, 10) having a list of at

least one related section (e.g. *section 1...n, dependency section 510* , Fig. 5); means for providing

the manifest with a hash of that related section (e.g. *Section 506*, Fig. 5 and *Hash 614*, Fig. 6;

next verdef section → structure #: hash value, Fig. 16).

But Evans does not specify that the related section is an assembly nor does Evans specify

a hash of a related assembly. The limitation of assembly being listed as opposed to section being

listed would have been obvious using Buxton's teaching as mentioned in claim 27 and the

limitation of providing a list of related assemblies or hash of such related assembly has also been

addressed using Buxton's teaching therein; hence is rejected herein using the corresponding

rejection of claim 27.

As per claims 31 and 32, Evans (with Buxton teaching from claim 27) further discloses

that one related section is a versioned object, i.e. module (re claim 31) composing a program

application executable (e.g. *Relocatable object 118, Shared object 114*, Fig. 2c-d; *versioned*

shared objects, col. 9, lines 20-22); that one related module is (re claim 32) being a referenced

section or module (e.g. *section 1...n, dependency section 510*, Fig. 5).

As per claim 33, Evans further discloses means to discern the difference between related

sections and/or versioned objects (col. 7, lines 29-56; Fig. 8,15; col. 14, lines 32-41; Fig. 18).

Refer to claim 6 for addressing the limitation about comparing hashes.

As per claim 34, Evans further discloses a binding policy, a limitation already set forth

in claim 26 and addressed therein.

As per claim 35, Evans further discloses that the related assembly or object is a

dynamically linked library (*libc* --col. 4, lines 20-29; *dynamic executable 120*, Fig. 2d).

Allowable Subject Matter

4.   Claims 9, 19-21 is objected to as being dependent upon a rejected base claim, but would

be allowable if rewritten in independent form including all of the limitations of the base claim

and any intervening claims.

Claims 10-17, 22 are allowed because of the feature recited as hash of a manifest, hash of

contents of a manifest of a referenced assembly or hash of a manifest of one referenced assembly

of the list of referenced assemblies.

Conclusion

5.   The prior art made of record and not relied upon is considered pertinent to applicant's

disclosure.

> U.S. Pat No. 6,308,320 to Burch, disclosing plurality of hash directory containing hash of objects file names.
>
> U.S. Pat No. 6,374,266 to Shnelvar, disclosing MDC structure containing hash values of objects in clusters.

Applicant's arguments with respect to claims 1-35 have been considered but are moot in

view of the new ground(s) of rejection.

Any inquiry concerning this communication or earlier communications from the

examiner should be directed to Tuan A Vu whose telephone number is (703)305-7207. The

examiner can normally be reached on 8AM-4:30PM/Mon-Fri.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's

supervisor, Kakali Chaki can be reached on (703)305-9662.

Any response to this action should be mailed to:

Commissioner of Patents and Trademarks

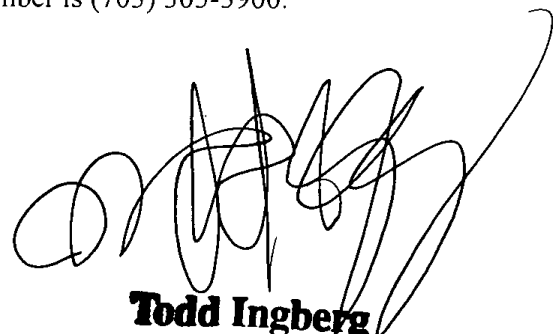Washington, D.C. 20231

or faxed to:

(703) 746-7239, (for formal communications intended for entry)

or: (703) 746-7240 (for informal or draft communications, please label

"PROPOSED" or "DRAFT")

Hand-delivered responses should be brought to Crystal Park II, 2121 Crystal Drive,

Arlington. VA. , 22202. 4th Floor(Receptionist).

Any inquiry of a general nature or relating to the status of this application or proceeding

should be directed to the receptionist whose telephone number is (703) 305-3900.

VAT
June 29, 2003

Todd Ingberg
Primary Examiner
Group 2100